Credit Rating with πGrammatical Evolution

Anthony Brabazon[1], **Michael O'Neill**[2]

[1] School of Business
University College Dublin, Ireland.
[2] Dept. Of Computer Science & Information Systems,
University of Limerick, Ireland.

Abstract. *This study examines the utility of πGrammatical Evolution in modelling the corporate bond-issuer credit rating process, using information drawn from the financial statements of bond-issuing firms. Financial data, and the associated Standard & Poor's issuer-credit ratings of 791 US firms, drawn from the year 1999/2000 are used to train and test the model. The best developed model was found to be able to discriminate in-sample (out-of-sample) between investment-grade and junk bond ratings with an average accuracy of 86 (87)% across a five-fold cross validation.*

1. Introduction The objective of this study is to determine whether a variant of Grammatical Evolution (GE) [1], πGE [2], is capable of accurately modelling the corporate bond rating process, using information drawn from the financial statements of bond-issuing firms. When a company issues traded debt (bonds), it must obtain a credit rating for the issue from at least one recognised rating agency (Standard and Poor's (S&P), Moody's and Fitches'). The credit rating represents an agency's opinion, at a specific date, of the creditworthiness of a borrower in general (a 'bond-issuer' credit-rating), or in respect of a specific debt issue (a 'bond' credit rating). These ratings impact on the borrowing cost, and the marketability of issued bonds. Although several studies have examined the potential of both statistical and machine-learning methodologies for credit rating prediction [3, 4, 5, 6], many of these studies used relatively small sample sizes, making it difficult to generalise strongly from their findings. This study by contrast, uses a large dataset of 791 firms, and introduces πGE to this domain.

In common with the related corporate failure prediction problem [7], a feature of the bond-rating problem is that there is no clear theoretical framework for guiding the choice of explanatory variables, or model form. Rating agencies assert that their credit rating process involves consideration of both financial and non-financial information about the firm and its industry, but the precise factors utilised, and the related weighting of these factors, are not publicly disclosed. In the absence of an underlying theory, most published work on credit rating prediction employs a data-inductive modelling approach, using firm-specific financial data as explanatory variables, in an attempt to 'recover' the model used by the rating agencies. This produces a high-dimensional combinatorial problem, as the modeller is attempting to uncover a 'good' set of model inputs, and model form, giving rise to particular potential for an evolutionary automatic programming methodology such as GE. In this initial application of πGE to modelling credit ratings, we restrict attention to the binary classification case, discriminating between investment grade vs junk grade ratings. The rest of this paper is organized as follows.

The next section provides a concise overview of Grammatical Evolution and its variant πGE, followed by a description of the data set and methodology utilised. The remaining sections provide the results of the study followed by a number of conclusions.

2. Grammatical Evolution Grammatical Evolution (GE) is an evolutionary algorithm that can evolve computer programs in any language

[1, 8, 9, 10], and can be considered a form of grammar-based genetic programming. Rather than representing the programs as parse trees, as in GP [11], a linear genome representation is used. A genotype-phenotype mapping is employed such that each individual's variable length binary string, contains in its codons (groups of 8 bits) the information to select production rules from a Backus Naur Form (BNF) grammar. The grammar allows the generation of programs in an arbitrary language that are guaranteed to be syntactically correct, and as such it is used as a generative grammar, as opposed to the classical use of grammars in compilers to check syntactic correctness of sentences. The user can tailor the grammar to produce solutions that are purely syntactically constrained, or they may incorporate domain knowledge by biasing the grammar to produce very specific forms of sentences. BNF is a notation that represents a language in the form of production rules. It is comprised of a set of non-terminals that can be mapped to elements of the set of terminals (the primitive symbols that can be used to construct the output program or sentence(s)), according to the production rules. A simple example BNF grammar is given below, where `<expr>` is the start symbol from which all programs are generated. The grammar states that `<expr>` can be replaced with either `<expr><op><expr>` or `<var>`. An `<op>` can become either `+`, `-`, or `*`, and a `<var>` can become either `x`, or `y`.

```
<expr> ::= <expr><op><expr> (0)
         | <var>            (1)
  <op> ::= +                (0)
         | -                (1)
         | *                (2)
 <var> ::= x                (0)
         | y                (1)
```

The grammar is used in a developmental process to construct a program by applying production rules, selected by the genome, beginning from the start symbol of the grammar. In order to select a production rule in GE, the next codon value on the genome is read, interpreted, and placed in the following formula:

$$Rule = Codon\ Value\ \%\ Num.\ Rules$$

where % represents the modulus operator. Given the example individual's genome (where each 8-bit codon has been represented as an integer for ease of reading) in Fig.1, the first codon integer value is 220, and given that we have 2 rules to select from for `<expr>` as in the above example, we get $220\ \%\ 2\ =\ 0$. `<expr>` will therefore be replaced with `<expr><op><expr>`.

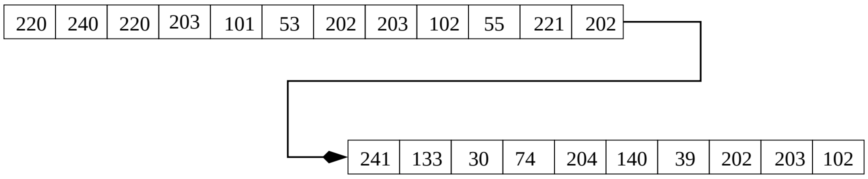
Beginning from the left hand side of the genome codon integer values are generated and used to select appropriate rules for the left-most non-terminal in the developing program from the BNF grammar, until one of the following situations arise:

- A complete program is generated. This occurs when all the non-terminals in the expression being mapped are transformed into elements from the terminal set of the BNF grammar.
- The end of the genome is reached, in which case the *wrapping* operator is invoked. This results in the return of the genome reading frame to the left hand side of the genome once again. The reading of codons will then continue unless an upper threshold representing the maximum number of wrapping events has occurred during this individual's mapping process.
- In the event that a threshold on the number of wrapping events has occurred and the individual is still incompletely mapped, the mapping process is halted, and the individual assigned the lowest possible fitness value.

Returning to the example individual, the left-most `<expr>` in `<expr><op><expr>` is mapped by reading the next codon integer value 240 and used in $240\ \%\ 2\ =\ 0$ to become another `<expr><op><expr>`. The developing program now looks like `<expr><op><expr><op><expr>`. Continuing to read subsequent codons and always mapping the left-most non-terminal the individual finally generates the expression `y*x-x-x+x`, leaving a number of unused codons at the end of the individual, which are deemed to be introns and simply ignored. A full description of GE can be found in [1].

3. πGrammatical Evolution The GE mapping process can be divided into a number of sub-components including the transcription and translation processes as outlined in the previous section. The πGE variant of GE replaces the translation process to allow evolution to specify the order in which production rules are mapped as op-

Fig. 1. An example GE individual's genome represented as integers for ease of reading.



posed to the strict depth-first, left to right, mapping of the standard GE algorithm. In πGE we use the genotype to dictate which non-terminal from those present to expand next, before deciding which production rule to apply to the selected non-terminal. The genome of an individual in πGE is different in that there are two components to each codon. That is, each codon corresponds to the pair of values ($nont$, $rule$).

In the first derivation step of the example mapping presented earlier, <expr> is replaced with <expr><op><expr>. Then in the standard GE genotype-phenotype mapping process, the left-most non-terminal (the first <expr>) in the developing program is always expanded first. The πGE mapping process differs in an individual's ability to determine and adapt the order in which non-terminals will be expanded [2]. To this end, a πGE codon corresponds to the pair $(nont, rule)$, where $nont$ and $rule$ are represented by N bits each (N=8 in this study), and a chromosome, then, consists of a vector of these pairs (see Figure 2). In πGE, we analyse the state of the developing program before each derivation step, counting the number of non-terminals present. If there is more than one non-terminal present in the developing program the next codon's $nont$ value is read to pick which non-terminal will be mapped next according to the following mapping function, where NT means non-terminal:

$$NT = Codon\ nont\ Value\ \%\ Num.\ NTs.$$

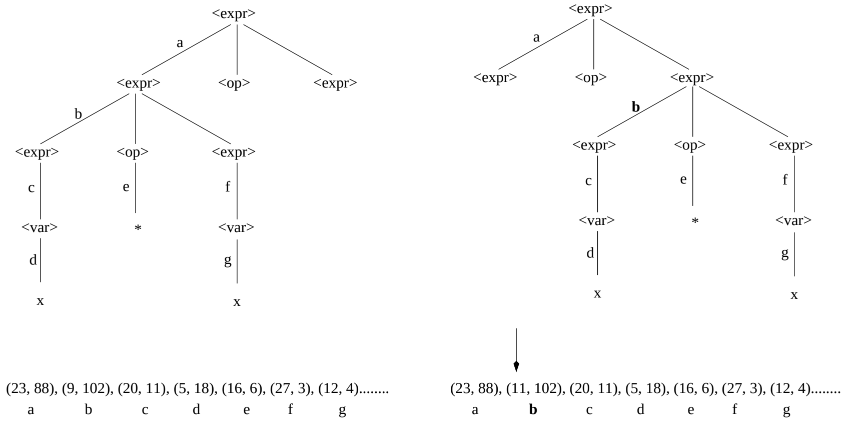
In the above example, there are 3 non-terminals (<expr>$_0$<op>$_1$<expr>$_2$) after application of the first production rule. To decide which non-terminal will be expanded next we use $NT = 9\ \%\ 3 = 0$, i.e., <expr>$_0$ is expanded. The mapping rule for selecting the appropriate rule to apply to the current non-terminal is given in the normal GE fashion:

$$Rule = Codon\ rule\ Value\ \%\ Num.\ Rules.$$

In this approach, evolution can result in a derivation subsequence being moved to a different context as when counting the number of non-terminals present we do not pay attention to the type of non-terminals (e.g. <expr> versus <op>). An alternative approach to πGE is to respect non-terminal types and only allow choices to be made between non-terminals of the same type, thus preserving the semantics of the following derivation subsequence, and simply changing the position in which it appears in the developing program.

4. Experimental Approach The dataset consists of financial data of 791 non-financial US companies, along with their associated bond-issuer credit-rating, drawn from the S&P Compustat database. Of these companies, 57% have an investment-grade rating (AAA, AA, A, or BBB), and 43% have a junk rating. To allow time for the preparation of year-end financial statements, the filing of these statements with the Securities and Exchange Commission (S.E.C), and the development of a bond rating opinion by Standard and Poor rating agency, the bond rating of the company as at 30 April 2000, is matched with financial information drawn from their financial statements as at 31 December 1999. A subset of 600 firms was randomly sampled from the total of 791 firms, to produce two groups of 300 'investment' grade and 300 'junk' rated firms. The 600 firms were randomly allocated to the training set (420) or the hold-out sample (180), ensuring that each set was equally balanced between investment and non-investment grade ratings. A total of eight financial variables were se-

Fig. 2. An example of piGE. In the left-hand derivation tree 9 Mod 3=0, hence the left-most non-terminal is expanded first. In the right-hand derivation tree 11 Mod 3=2, hence the right-most non-terminal is expanded.



lected for inclusion in this study. The selection of these variables was guided both by prior literature in bankruptcy prediction [12], literature on bond rating prediction [13], and by preliminary statistical analysis. The objective in selecting a set of proto-explanatory variables is to choose financial variables that vary between companies in different bond rating classes, and where information overlaps between the variables are minimised (the financial ratios chosen during the selection process are listed at the end of this section). Comparing the means of the chosen ratios for the two groups of ratings, reveals a statistically significant difference at both the 5% and the 1% level, and as expected, the financial ratios in each case, for the investment ratings are stronger than those for the junk ratings. The only exception is the current ratio, which is stronger for the junk rated companies, possibly indicating a preference for these companies to hoard short-term liquidity, as their access to long-term capital markets is weak. A correlation analysis between the selected ratios indicates that most of the cross-correlations are less than $\mid 0.20 \mid$, with the exception of the debt ratio and (Retained Earnings/Total Assets) ratio pairing, which has a correlation of -0.64. The grammar adopted is as follows:

```
<lc> ::= if( <expr> <relop> <expr> ) class=''Junk'';
 else class=''Investment Grade'';

<expr> ::= ( <expr> ) + ( <expr> ) | <coeff> * <var>

<var> ::= var3[index] | var4[index] | var5[index]
        | var6[index] | var7[index] | var8[index]
        | var9[index] |var10[index] | var11[index]

<coeff> ::=  ( <coeff> ) <op> ( <coeff> ) | <float>

<op> ::= + | - | *

<float> ::= 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2
 | 1 | -1 | .1

<relop> ::= <=
```

where `var3` = Current Ratio, `var4` = Retained Earnings to total assest, `var5` = Interest Coverage, `var6` = Debt Ratio, `var7` = Net Margin, `var8` = Market to book value, `var9` = Total Assets, `var10` = ln (Total Assets), `var11` = Return on total assets.

5. Results The results from our experiments are now provided. Each of the πGE experiments is run for 100 generations, with variable-length, one-point crossover at a probability of 0.9, one point bit mutation at a probability of 0.01, roulette selection, and steady-state replacement. To assess the stability of the results across different randomisations of the dataset between training and test data, we recut the dataset five times, maintaining an equal balance of investment and non-investment grade ratings in the resulting training and test datasets. In our experiments, fitness is defined as the number of correct classifications obtained by an evolved discriminant rule. The results for the best individual of each cut of the dataset, where 30 independent runs were performed for each cut, averaged over all five randomisations of the dataset, for a population size of 500 is given in Table 1, and Figure 3 displays the evolution of the mean average and mean best results over time.

Table 1. Average performance for the five recuts of the best evolved rules on their in and out-sample datasets.

	Fitness	TP	TN	FP	FN
In-sample	0.8450	182.8	172.1	37.9	27.2
Out-sample	0.8500	77.9	75.1	14.9	12.1

To assess the overall hit-ratio of the developed models (out-of-sample), Press's Q statistic [14] was calculated for each model. In all cases, the null hypothesis, that the out-of sample classification accuracies are not significantly better than those that could occur by chance alone, was rejected at the 1% level. A t-test of the hit-ratios also rejected a null hypothesis that the classification accuracies were no better than chance at the 1% level. Across all the data recuts, the best individual achieved an 86 (87)% accuracy in-sample (out-of-sample). Examining the structure of the best individuals shows that the evolved discriminant functions had the following form:

IF (0 $\leq$ -2 + var6 - var9 - 5*var4) THEN 'Junk' ELSE 'Investment Grade'

where var4 is $\frac{Retained\ Earnings}{Total\ Assets}$, var6 is $Debt\ Ratio$, and var9 is $Total\ Assets$.

Examining the signs of the coefficients of the evolved rules does not suggest that they conflict with common financial intuition. The rules indicate that low/negative retained earnings, low/negative total assets or high levels of debt finance are symptomatic of a firm that has a junk rating. It is noted that similar risk factors have been identified in predictive models of corporate failure which utilise financial ratios as explanatory inputs [7, 15]. Conversely, low levels of debt, a history of successful profitable trading, and high levels of total assets are symptomatic of firms that have an investment grade rating.

5.1. Comparison of Results To provide a benchmark for the results obtained by πGE we compare them with the results obtained on the same recuts of the dataset, using a fully-connected, feedforward multi-layer perceptron (MLP), trained using the backpropagation algorithm. The developed networks utilised all the explanatory variables. The optimal number of hidden-layer nodes was found following experimentation on each separate data recut, and varied between two and four nodes. The classification accuracies for the networks, averaged over all five recuts is provided in Table 2.

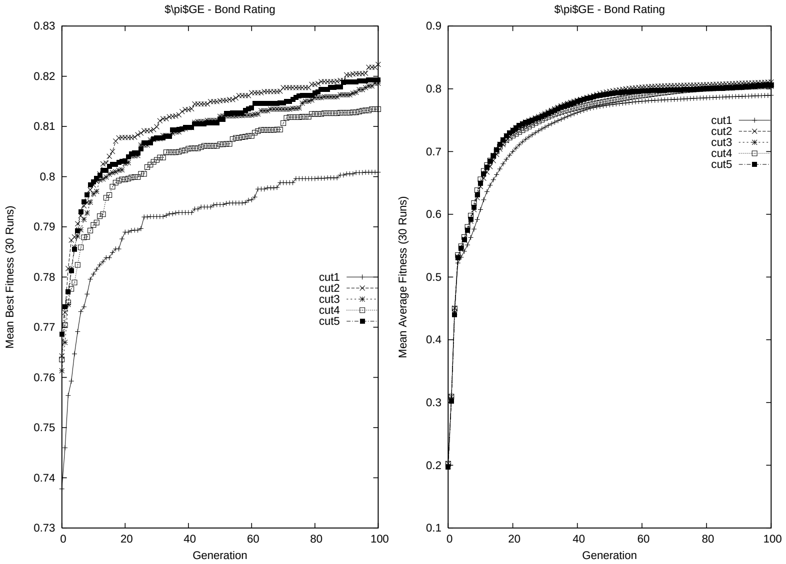
Table 2. Performance of the MLPs on the training and out-of-sample datasets, averaged over all five recuts of the dataset.

	Fitness	TP	TN	FP	FN
In sample	0.869	181.8	183.2	26.8	28.2
Out-sample	0.850	75.8	77.2	12.8	14.2

The levels of classification accuracy obtained with the MLP are competitive with earlier research, with for example [13] obtaining an out-of-sample classification accuracy of approximately 83.3%, although it is noted that the size of the dataset in their study was small. Comparing the results from the MLP with those of πGE on the initial fitness function (Table 1) suggests that πGE has proven competitive with an MLP methodology, in terms of producing a similar classification accuracy. Benchmark results were also obtained using an LDA methodology. Utilising the same dataset recuts as πGE, LDA produced results (averaged across all five recuts) of 82.74% in-sample, and 85.22% out-of-sample. Again, πGE is competitive against these results in terms of classification accuracy. Comparing the results obtained by the linear classifiers (LDA and πGE) against those of an MLP, suggests that strong non-linearities between the explanatory variables and the dependent variable are not present.

6. Conclusions & Future Work The objective of this paper was to assess the utility of the πGE methodology to model the corporate bond rating process, using information drawn from the financial statements of bond-issuing firms. Despite us-

Fig. 3. Mean average (left) and mean best (right) over 30 runs, across all 5 recuts.



ing data drawn from companies in a variety of industrial sectors, the developed models showed an impressive capability to discriminate between investment and junk rating classifications. The πGE developed models also proved highly competitive with a series of MLP models developed on the same datasets. Several extensions of the methodology in this study are indicated for future work. One route is the inclusion of non-financial company and industry-level information as input variables. A related possibility would be to concentrate on building rating models for individual industrial sectors. We are also extending this study to encompass the multi-class rating prediction problem.

References.

[1] O'Neill M., Ryan C. (2003) *Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language*, Kluwer Academic Publishers.

[2] O'Neill M., Brabazon A., Nicolau M., Mc Garraghy S. and Keenan P. (2004). πGrammatical Evolution. In *Proceedings of GECCO 2004*, Seattle, WA, USA.

[3] Ederington, H. (1985). 'Classification models and bond ratings', *Financial Review*, 20(4):237-262.

[4] Gentry, J., Whitford, D. and Newbold, P. (1988). 'Predicting industrial bond ratings with a probit model and funds flow components, *Financial Review*, 23(3):269-286.

[5] Huang, Z., Chen, H., Hsu, C., Chen, W. and Wu, S. (2003). 'Credit rating analysis with support vector machines and neural networks: a market comparative study', *Decision Support Systems*, (Article in press).

[6] Shin, K. and Han, I. (2001). 'A case-based approach using inductive indexing for corporate bond rating', *Decision Support Systems*, 32:41-52.

[7] Brabazon, T., O'Neill, M., Matthews, R., and Ryan, C. (2002). Grammatical Evolution and Corporate Failure Prediction, In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002)*, Spector et. al. Eds., July 9-13, 2002, pp. 1011-1019, New York: Morgan Kaufmann.

[8] O'Neill, M. (2001). Automatic Programming in an Arbitrary Language: Evolving Programs in Grammatical Evolution. PhD thesis, University of Limerick.

[9] O'Neill M., Ryan C. (2001) Grammatical Evolution, *IEEE Trans. Evolutionary Computation*. 2001.

[10] Ryan C., Collins J.J., O'Neill M. (1998). Grammatical Evolution: Evolving Programs for an Arbitrary Language. *Lecture Notes in Computer Science 1391, Proceedings of the First European Workshop on Genetic Programming*, pp. 83-95, Springer-Verlag.

[11] Koza, J. (1992). *Genetic Programming*, MIT Press.

[12] Altman, E. (1993). *Corporate Financial Distress and Bankruptcy*, New York: John Wiley and Sons Inc.

[13] Dutta, S. and Shekhar, S. (1988). 'Bond rating: a non-conservative application of neural networks, *Proceedings of IEEE International Conference on Neural Networks*, II, 443-450.

[14] Hair, J., Anderson, R., Tatham, R. and Black, W. (1998). *Multivariate Data Analysis*, Upper Saddle

River, New Jersey: Prentice Hall.

[15] Brabazon, T. and O'Neill, M. (2003). Anticipating Bankruptcy Reorganisation from Raw Financial Data using Grammatical Evolution, Proceedings of EvoIASP 2003, *Lecture Notes in Computer Science (2611): Applications of Evolutionary Computing*, edited by Raidl, G., Meyer, J.A., Middendorf, M., Cagnoni, S., Cardalda, J. J. R., Corne, D. W., Gottlieb, J., Guillot, A., Hart, E., Johnson, C. G., Marchiori, E., pp. 368-378, Berlin: Springer-Verlag.